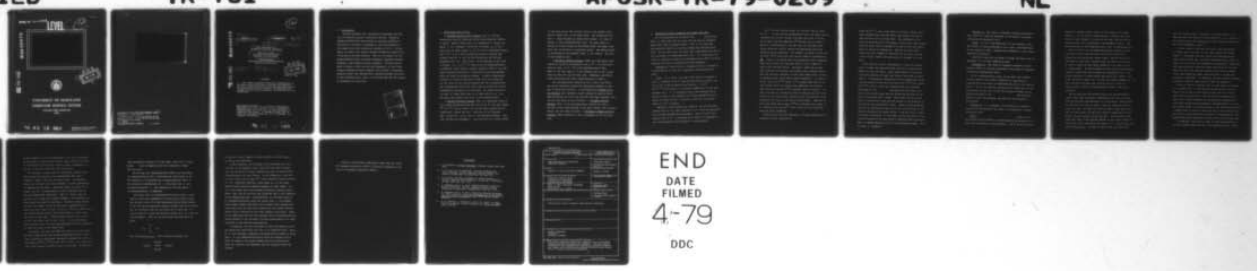
AD-A065 675 MARYLAND UNIV COLLEGE PARK COMPUTER SCIENCE CENTER F/G 9/2
SOME PROPERTIES OF BOTTOM-UP CELLULAR PYRAMIDS.(U)
FEB 79 C R DYER , A NAKAMURA AFOSR-77-3271A
UNCLASSIFIED TR-731 AFOSR-TR-79-0209 NL

| OF |
ADA
065675



END
DATE
FILMED
4-79
DDC

LEVEL II

12

UNIVERSITY OF MARYLAND
COMPUTER SCIENCE CENTER

COLLEGE PARK, MARYLAND
20742

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFSC)
NOTICE OF TRANSMITTAL TO DDC
This technical report has been reviewed and is approved for public release IAW AFR 190-12 (7b).
Distribution is unlimited.
A. D. BLOSE
Technical Information Officer

(14) TR-731
(15) AFOSR-77-3271A
(11) February 1979
(12) 19 p.
(9) Technical rept.

(6) SOME PROPERTIES OF BOTTOM-UP CELLULAR PYRAMIDS

(10) Charles R. Dyer
Computer Science Center
University of Maryland

Akira Nakamura
Department of Applied Mathematics
Hiroshima University

(18) AFOSR
(19) TR-79-0209

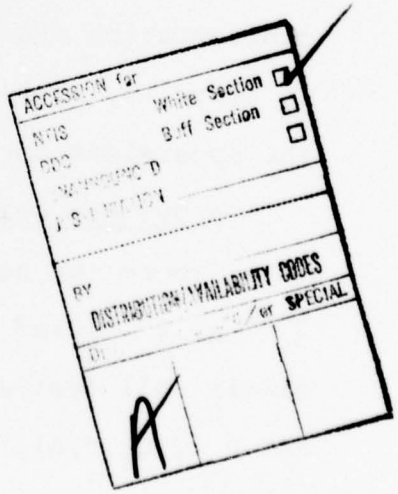
ABSTRACT

The formal language recognition capabilities of bottom-up pyramid cellular acceptors are examined. The main result establishes that deterministic bottom-up pyramid acceptors are weaker than deterministic bounded cellular array acceptors, in both one and two dimensions.

The support of the U.S. Air Force of Scientific Research under Grant AFOSR-77-3271 is gratefully acknowledged, as is the help of Dawn Shifflett in preparing this paper. A. Nakamura would like to thank his research assistant, Dr. T. Watanabe, for valuable discussions.

1. Introduction

Cellular pyramids were introduced by Rosenfeld and Dyer [1,2] as parallel pattern recognition devices. Bottom-up pyramid acceptors have been defined as cellular pyramids restricted in information transmission, and their properties and capabilities were extensively studied in [1-5]. In this paper, we examine the formal language recognition capabilities of bottom-up cellular pyramids. That is, we compare the language-accepting power of pyramid acceptors, bottom-up pyramid acceptors, and bounded cellular acceptors. The main result, which is shown in both one and two dimensions, is that deterministic bounded cellular acceptors are stronger in language-accepting power than deterministic bottom-up pyramid acceptors. In the nondeterministic case, it is proved that the two classes of languages are equivalent.



2. Definitions and notation

A bounded cellular array acceptor (CA) is a finite, rectangular array of identical finite state machines (FSM's), or cells. Each of these cells is a quadruple $M = (Q, Q_I, \delta, A)$, where Q is a nonempty, finite set of states, $Q_I \subseteq Q$ is a finite set of input states, $A \subseteq Q$ is a set of accepting states, and $\delta: Q^5 \to Q$ is a state transition function, mapping the current state of M and its four horizontal and vertical neighbors into M's next state. If the mapping is into sets of states, i.e., $\delta: Q^5 \to 2^Q$, then the CA is nondeterministic. In addition, there exists a special boundary state $\# \in Q_I$. The state transition function is restricted so that the boundary state can never be exited or entered. A step of computation consists of a state transition of each cell; the states of all the cells at any time step define the CA's configuration. The configuration before the first step is called the initial configuration and must be of the form: all border cells are in state #, every other cell is in some state in $Q_I - \{\#\}$. The upper-left corner non-# cell is the accepting cell.

A pyramid cellular acceptor (PCA) is a pyramidal stack of CA's, where the bottom array has size 2^n by 2^n, the next lowest 2^{n-1} by 2^{n-1}, and so forth, the (n+1)st layer consisting of a single cell, called the root. Each cell is an identical FSM = (Q, Q_I, δ, A). Q, Q_I, and A are defined as before. Each cell now has nine neighbors -- four son cells in a 2-by-2 block

in the level below, four brother cells in the current level, and one father cell in the level above. The transition function δ maps 10-tuples of states into states -- or sets of states, in the nondeterministic case. An input array is stored as initial states of the bottom array; the upper-level cells are initialized to a quiescent state. The whole pyramid is surrounded by the boundary state # as before. The root is the accepting cell.

A bottom-up pyramid acceptor (UPCA) is a PCA whose state transition function is modified to be $\delta: Q^5 \rightarrow Q$ in the deterministic case, $\delta: Q^5 \rightarrow 2^Q$ in the nondeterministic case. That is, the next state of a cell depends only on the current state of that cell and its four sons. Otherwise, the definitions associated with PCA's extend directly to UPCA's.

The one-dimensional analogs of CA's, PCA's, and UPCA's are easily defined by making the appropriate changes in each cell's transition function. A one-dimensional bounded cellular acceptor (CA) contains a state transition function which maps the states of a cell and its left and right neighbors into its own next state or set of states. A triangle cellular acceptor (TCA) is the one-dimensional version of a PCA; a TCA cell's next state depends on two sons' states, two brothers' states, and its father's state. A bottom-up triangle cellular acceptor (UTCA) restricts a cell's neighbors to its two sons only.

3. Bottom-up triangle acceptors are weaker than CA's

In this subsection we establish that restricting TCA's, so that each cell has only its two sons as neighbors, diminishes their acceptance power. That is, we show that deterministic UTCA's are strictly weaker than deterministic CA's. On the other hand, we prove that nondeterministic UTCA's are equivalent to nondeterministic CA's. It follows that the class of languages accepted by nondeterministic UTCA's strictly contains the class of determinstc UTCA languages.

Theorem 3.1 There exists a language which is accepted by a deterministic CA, but not accepted by any deterministic UTCA.

Proof: It is known [6] that there exists a language L which is accepted by an n^2-tape bounded Turing acceptor, but not by any n-tape bounded Turing acceptor. Let $L' = \{\sigma b^{|\sigma|^2-|\sigma|} \mid \sigma \in L\}$, where b is a special symbol not in the tape alphabet of L. L' is accepted by an n-tape bounded Turing acceptor since it can simulate an n^2-tape bounded Turing acceptor on input string σ, and also verify that the proper number of b's is present.

Suppose L' were accepted by a UTCA M. We now show that because most of its input consists of b's, M can be simulated by a $\sqrt{n}$-tape bounded Turing acceptor. Since $|\sigma|=\sqrt{n}$, this implies that if L' is accepted by M, then L is accepted by an n-tape bounded Turing acceptor--a contradiction.

Let 2^n be the smallest power of 2 greater than or equal to $|\sigma|$. In the initial configuration of M, the input string σ defines the initial states of at most the first 2^n cells of the base, and the remaining $2^{2n}-2^n$ cells are in state b. Since M is deterministic, any two cells at the same level having identical initial subtree configurations must have identical state sequences. For example, at level 0 there are at least $2^{2n}-2^n$ cells with the initial subtree configuration  . Since all of these cells have identical state sequences, there are at most 2^n+1 distinct level 0 cells, namely 2^n non-b cells and one b-cell. (Actually, there are at most $|Q_I|$ distinct non-b cells, but because their ancestor cells may have distinct subtrees, we must save these cells in order to compute higher level cells' state sequences.) At level k, $0 \leq k \leq n$, there are 2^{2n-k} cells, 2^{n-k} of which have non-b's in their bases, while the others have identical all-b bases. Thus there are at most $2^{n-k}+1$ distinct cells at level k. At each of levels n+1 through 2n there is exactly one cell with non-b's in its base; hence there are just two distinct state sequences for cells at any one of these levels. Summing, we find that there are at most $2^{n+1}+3n-1$ distinct cell types in M. Thus while M has $O(2^{2n})$ cells, only $O(2^n)$ of them have distinct state sequences.

Based on this, we now construct a 2^n-tape bounded Turing acceptor T which simulates M.

Order the $2^{n+1}-1$ cells from levels 0 through n having non-b bases breadth-first and map their input states into the 2^n tape positions of T, two per cell. The remainder of M's cells to be saved are of two types: 2n cells with all-b bases, one each from levels 0 through 2n-1; and n cells with non-b bases at levels n+1 through 2n. Map the 2n all-b cells' states into the leftmost 2n positions of T's tape, and the n non-b cells' states into positions n+1 through 2n of the tape.

Given such an initial configuration, T acts as follows. First, T marks off tape positions n and 2n by counting the number of tape positions. Using these marks, T can easily find those sections of tape that contain the extra 3n cells' states. T simulates a transition of M by sequentially accessing the states of the sons for each of M's $2^{2n+1}+3n-1$ cells. The $2^{n+1}+1$ cells' states are changed using the technique described in [2]. The states of both sons of an all-b cell are the same, and are stored one position to the left of the given cell's state. The states of each of the other n non-b cells' sons are both stored in the position to the left of the given cell's state. (There is one exception: the cell at position n+1 has as its left son the root of the subtree containing the input string σ, and this cell's state is stored in position 1.) The state of the root cell of M is stored in position 2n; thus after T completes each simulation step, it checks whether or not it is an accepting state. If it is, then T accepts.//

Theorem 3.2 The class of languages accepted by deterministic UTCA's is strictly contained in the class of languages accepted by deterministic CA's.

Proof: In [2] it was shown how a CA can simulate a TCA. That result is easily modified to prove that a one-dimensional CA can simulate a UTCA. Together with Theorem 3.1, this result immediately follows.//

Since TCA's and CA's are known to accept the same class of languages [3], we also immediately have

Corollary 3.1 The class of languages accepted by deterministic UTCA's is strictly contained in the class of languages accepted by deterministic TCA's.

In the nondeterministic case, we now show that nondeterministic UTCA's (for brevity: NUTCA's) are equivalent in language-accepting power to nondeterministic CA's (for brevity: NCA's). First, any language accepted by an NUTCA can also be accepted by an NCA since an NCA can simulate an NUTCA by the method described in [2]. The following theorem proves the converse, and thus the relationship is established.

Theorem 3.3 If a language L is accepted by a nondeterministic or deterministic CA, then L is accepted by a nondeterministic UTCA.

Proof: Given a CA A, we describe how an NUTCA M is constructed which simulates A in real time (following a startup delay). Let 2^n be the smallest

power of 2 greater than or equal to the length of the input string σ. In M's initial configuration, σ defines the input states of the leftmost $|\sigma|$ cells of the base, and the remaining base cells are initialized to the boundary state #. The generalization to the nondeterministic case is straightforward.

Each base cell in M nondeterministically chooses at each time step a state from the state set Q of A, while also remembering its previously chosen state. Thus at the end of time step t, each cell c stores a pair of states (p,q) from Q, where p and q are the states chosen by c at times t-1 and t, respectively. To check whether or not the new configuration legally follows from the previous one according to A's transition function δ, M must verify that $q \in \delta(r,p,s)$, where r and s are the states chosen by c's left and right neighbors at t-1. This involves verifying that (r,u)(p,q)(s,v) is one of a finite number ($<|Q|^6$) of legal patterns, where u and v are arbitrary.

We now show how the non-base cells in M can deterministically check that the base's configuration at time t follows from the previous configuration in n time steps. Specifically, at time step t+1 cells at level 1 copy the state pairs from their two sons. At time t+2 cells at level 2 copy their sons' pair of state pairs; so each cell stores a quadruple of state pairs $((p_1,q_1),(p_2,q_2),(p_3,q_3),(p_4,q_4))$. At the end of this step each level 2 cell detects whether or not their second and third base cells' new states are legal, i.e., if $q_2 \in \delta(p_1,p_2,p_3)$ and $q_3 \in \delta(p_2,p_3,p_4)$. If both of these tests are true, then

the cell enters state t, otherwise, it enters state f. In addition each cell checks its end conditions, i.e., whether or not $q_1 \in \delta(\#, p_1, p_2)$ and $q_4 \in \delta(p_3, p_4, \#)$, and stores the results in two state variables, ℓ and r.

At time step t+3 each cell at level 3 acts as follows. If either son is in state f, then the current cell enters state f. Otherwise, the cell copies the third and fourth pairs from its left son, and first and second pairs from its right son to form its own quadruple of state pairs. These are just the state pairs from the middle four cells in its base. Thus this cell next verifies whether or not the two middle cells' new states are legal, entering state t if they are, and state f otherwise. In addition, the cell copies the state variable ℓ from its left son and r from its right son.

Cells at levels 4 through n act at time steps t+4 through t+n, respectively, in the same way as the cells at level 3. Thus if at the end of time step t+n the root is in state t and ℓ and r are both true, then the base's configuration at time t was a legal successor to the previous configuration. Since this verification procedure is accomplished by a single unit speed "wave" of activity moving up the tree involving only a single level of cells at each time step, the non-base cells can verify in real time the succession of configurations non-deterministically entered by the base cells at each time step.

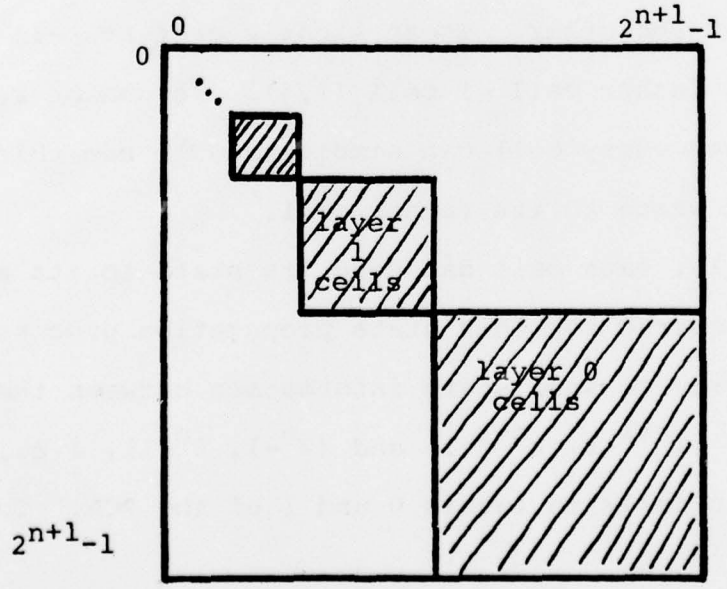
In addition, the root can check at each step whether or not the leftmost base cell was in an accepting state. Hence

if A goes through a sequence of configurations leading to acceptance, M can nondeterministically guess the sequence, check its legality, and determine that an accepting state was entered. Furthermore, the simulation is real time following a log diameter time start up delay.//

4. Bottom-up pyramid acceptors are weaker than CA's

Theorem 3.2, which proved that one-dimensional CA's are stronger than UTCA's, can be generalized to two dimensions. In this case a UPCA with base size 2^{2n} by 2^{2n}, and all blank input except in the top row, can be simulated by a 2^{2n}-tape bounded Turing acceptor since the UPCA has at most $2^{2n+2}/3 + 3n$ distinct state sequences for its cells. We show below that a two-dimensional CA can simulate a PCA or UPCA; the desired result now follows from the fact that a 2^{4n}-tape bounded Turing acceptor is strictly stronger than a 2^{2n}-tape bounded Turing acceptor.

A CA of size 2^{n+1} by 2^{n+1}, or equivalently, one of size 2^n by 2^n with each cell storing a 2 by 2 block of states, can simulate a PCA with base size 2^n by 2^n as follows. Map the PCA cells into the CA cells as shown in the following diagram.



In this mapping a cell at coordinates (i,j) has its brothers in positions (i-1,j),(i+1,j),(i,j-1), and (i,j+1); its father is in position ($\lfloor i/2 \rfloor, \lfloor j/2 \rfloor$), and its sons's coordinates are (2i,2j), (2i+1,2j), (2i,2j+1), and (2i+1,2j+1).

To simulate a single step of a given PCA, each CA cell must access the states of its corresponding PCA cell's brothers, father, and sons (if they exist). Its brothers' states are stored by its four neighbors, so this information is immediately available. Each cell sends its state to its father cell by a generalization of the technique used for the one dimensional simulation. That is, first a cell at position (i,j) sends two signals leftward, one travelling at three times the speed of the other. The fast signal bounces off the left border of the CA and moves rightward until it meets the slow signal at cell ($\lfloor i/2 \rfloor$,j). This cell then starts two signals moving upward, again one moving at three times the speed of the other. These signals meet at cell ($\lfloor i/2 \rfloor$, $\lfloor j/2 \rfloor$), the father cell of cell (i,j). It can be readily verified that every cell can simultaneously use this procedure to send its state to its father cell.

Similarly, each cell can send its state to its son cells. The time required for this state propagation process is the time necessary to send state information between the cells at coordinates ($2^{n+1}-1$, $2^{n+1}-1$) and (2^n-1, 2^n-1), i.e., the lower-right corner cells in levels 0 and 1 of the PCA. To determine

each coordinate requires $3 \cdot 2^n$ time steps, hence $6 \cdot 2^n = O(\text{diameter})$ time is needed by the CA to simulate a single PCA time step.

We now show that nondeterministic UPCA's are equivalent to nondeterministica CA's, correcting the proof given in [4]. For simplicity, we describe how a nondeterministic UPCA M can simulate a deterministic CA A with state set Q and transition function δ. The extension to the case where A is nondeterministic is immediate.

Each base cell in M nondeterministically enters a state from Q, while also remembering its previously chosen state. The non-base cells of M then deterministically check whether or not the new configuration legally follows from the previous one, by verifying that the new state (q) of each cell c is in the range of δ given the previous states (p's) of c and its four neighbors. That is, we must verify for each block of cells

$$\begin{array}{ccc} & c_n & \\ c_w & c & c_e \\ & c_s & \end{array}$$

that $q \in \delta(p, p_n, p_w, p_e, p_s)$. This involves checking that

$$\begin{array}{ccc} & (p_n, q_n) & \\ (p_w, q_w) & (p, q) & (p_e, q_e) \\ & (p_s, q_s) & \end{array}$$

is one of a finite number of legal patterns (in which q_n, q_w, q_e and q_s are arbitrary).

In one dimension, the non-base cells performed this verification in log diameter time, since each cell had to check just the two blocks of three consecutive pairs of states which crossed between its sons' bases. In two dimensions, each cell in level k must verify that 2^{k+2} local patterns follow according to A's transition function, since there are now this many patterns which cross the borders between its sons' bases. In [3] it was shown that local property detection requires O(diameter) time, and so the real time algorithm used in one dimension is not appropriate here. Alternatively, let the base cells of M nondeterministically enter new states from Q at nondeterministically chosen time steps; the non-base cells deterministically check that all base cells chose their new states synchronously and at intervals no less than diameter time steps. There is now sufficient time for the non-base cells to deterministically verify whether or not the base's current configuration is a legal successor to the previous configuration.

In addition, the root can check at each step whether or not the upper-left corner base cell was in an accepting state. Hence if A goes through a sequence of configurations leading to acceptance, M can nondeterministically guess the sequence with a delay of diameter time steps between each new configuration, check its legality, and determine that an accepting state was entered.

Finally, these results immediately imply that the class of languages accepted by UPCA's is strictly contained in the class of languages accepted by NUPCA's.

References

1. A. Rosenfeld, Picture Languages, Academic Press, New York, 1979.

2. C. R. Dyer and A. Rosenfeld, Cellular pyramids for image analysis, TR-544, Computer Science Center, University of Maryland, College Park, MD, May 1977.

3. C. R. Dyer, Cellular pyramids for image analysis, 2, TR-596, Computer Science Center, University of Maryland, College Park, MD, Nov. 1977.

4. A. Nakamura and C. R. Dyer, Nondeterministic bottom-up pyramid acceptors, TR-616, Computer Science Center, University of Maryland, College Park, MD, Dec. 1977.

5. A. Nakamura and C. R. Dyer, Bottom-up cellular pyramids for image analysis, Proc. 4th International Joint Conference on Pattern Recognition, Nov. 1978, Kyoto, Japan, 494-496.

6. R. E. Stearns, J. Hartmanis, and P. M. Lewis II, Hierarchies of memory limited computations, Proc. 6th SWAT, 1965, 179-190.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) SOME PROPERTIES OF BOTTOM-UP CELLULAR PYRAMIDS		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER TR-731
7. AUTHOR(s) Charles R. Dyer and Akira Nakamura		8. CONTRACT OR GRANT NUMBER(s) AFOSR-77-3271A
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computer Science Center University of Maryland College Park, MD 20742		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Math. & Info. Sciences, AFOSR/NM Bolling AFB Washington, D.C. 20332		12. REPORT DATE February 1979 13. NUMBER OF PAGES 17
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Formal languages
Acceptors
Cellular automata

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The formal language recognition capabilities of bottom-up pyramid cellular acceptors are examined. The main result establishes that deterministic bottom-up pyramid acceptors are weaker than deterministic bounded cellular array acceptors, in both one and two dimensions.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)